

EBERHARD KARLS

UNIVERSITÄT
TÜBINGEN



Technische Informatik

Basispraktikum Sommersemester 2001

Protokoll zum Versuchstag 6

Datum: 5.7.2001

Gruppe: David Eißler/ Marc-Oliver Pahl

Autor: Marc-Oliver Pahl



Verwendete Messgeräte:

- Minirechner (MR1)
- Netzgerät (NG2)

Versuch 2: Rechnen

a) $(15 - 3) / 4$

Befehl					Adresse							Bit	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
SRG B	SRG A	MUX	ALU	SRG F								IC	SRG B		SRG A				MUX	SRG F		ALU								
					8	7	6	5	4	3	2	1	0	Pin	S1	S0	SIL	SIR	S1	S0	SEL	S1	S0	-Cn	M	S3	S2	S1	S0	
LD3	LD15	A	A-B	LD								0		I	I				I	I	0	I	I	0	0	0	0	I	I	0
				SHR								I									I	0								
				SHR								I	0								I	0								

b) $(6 * 2 - 9) * 2$

Befehl					Adresse							Bit	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
SRG B	SRG A	MUX	ALU	SRG F								IC	SRG B		SRG A				MUX	SRG F		ALU							
					8	7	6	5	4	3	2	1	0	Pin	S1	S0	SIL	SIR	S1	S0	SEL	S1	S0	-Cn	M	S3	S2	S1	S0
	LD6											0																	
	SHL0											I							0	0	I								
LD9		A	A-B	LD								I	0	I	I						0	I	I	0	0	0	I	I	0
				SHL								I	I								0	I							

c) $((4 * 2 - 7) * 2 + 6) / 4$

Befehl					Adresse							Bit	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
SRG B	SRG A	MUX	ALU	SRG F								IC	SRG B		SRG A				MUX	SRG F		ALU							
					8	7	6	5	4	3	2	1	0	Pin	S1	S0	SIL	SIR	S1	S0	SEL	S1	S0	-Cn	M	S3	S2	S1	S0
	LD4											0																	
	SHL											I						0	0	I									
LD7		A	A-B	LD								I	0	I	I						0	I	I	0	0	0	I	I	0
				SHL								I	I								0	I							
LD6		F	A+B	LD								I	0	I	I					I	I	I	I	0	I	0	0	I	
				SHR								I	0	I							I	0							
				SHR								I	0	I							I	0							

d) $(5 - (1 + 5) / 2) * 4$

Befehl					Adresse							Bit	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
SRG B	SRG A	MUX	ALU	SRG F								IC	SRG B		SRG A				MUX	SRG F		ALU							
					8	7	6	5	4	3	2	1	0	Pin	S1	S0	SIL	SIR	S1	S0	SEL	S1	S0	-Cn	M	S3	S2	S1	S0
LD5	LD1	A	A+B	LD										I	I					I	I	0	I	I	I	0	I	0	0
				SHR									I								I	0							
		F	-A	LD								I	0								I	I	I	I	I	0	0	0	0
		F	A+B+1	LD								I	I								I	I	I	0	0	I	0	0	
				SHL								I	0	0							0	I							
				SHL								I	0	I							0	I							

Es gibt keinen ALU-Befehl B - A, deshalb wird die Subtraktion durch Addition des Zweierkomplements (=Einerkomplement+1) ersetzt...



Versuch 3: Division durch 7

Unser Algorithmus, der die richtigen Ergebnisse für die ganzzahlige Division im Bereich 0-15 abdeckt ist $1r(2r(A) + 1)$.

Die Funktionstabelle für die damit gültigen Werte ist:

<u>Eingabe (A)</u>	<u>Ausgabe ($1r(2r(A) + 1)$)</u>	
1 1 1 0	0 0 1 0	(14/7 = 2)
0 1 1 1	0 0 0 1	(7/7 = 1)
0 0 0 0	0 0 0 0	(0/7 = 0)

Befehl				Adresse								Bit	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
SRG B	SRG A	MUX	ALU	SRG F									IC	SRG B		SRG A				MUX	SRG F		ALU							
					8	7	6	5	4	3	2	1	0	Pin	S1	S0	SIL	SIR	S1	S0	SEL	S1	S0	-Cn	M	S3	S2	S1	S0	
	LD14	A	A	LD									0									0	1	1	1	1	1	1	1	
				SHR									1									1	1	0						
				SHR								1	0									1	1	0						
		F	A+1	LD									1	1							1	1	1	1	0	1	0	0	1	
				SHR								1	0	0							1	1	0							

Versuch 4: Multiplikation zweier 4Bit-Zahlen

Der Algorithmus funktioniert fast so, wie man ihn auch von Hand auf dem Blatt durchführen würde, nur werden die Stellkeitsverschiebungen erst im Nachhinein aufgebaut.

$$\begin{array}{r}
 1100 * 1010 \\
 \hline
 1100000 \\
 + 11000 \\
 \hline
 1111000
 \end{array}$$

Befehl				Adresse								Bit	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
SRG B	SRG A	MUX	ALU	SRG F									IC	SRG B		SRG A				MUX	SRG F		ALU								
					8	7	6	5	4	3	2	1	0	Pin	S1	S0	SIL	SIR	S1	S0	SEL	S1	S0	-Cn	M	S3	S2	S1	S0		
	LD	LD											0		1	1															
				NOP									1						0	0	1										
	SHL		B	LD	1								1						0	0	1										
				SHL									1	0									0	1							
				SHL	1								1	0									0	1							
	SHL			NOP									1	1						0	0	1									
	SHL	F	A+B	LD	1								1	1						0	0	1									
				SHL									1	0	0									0	1						
				SHL	1								1	0	0									0	1						
	SHL			NOP									1	0	1						0	0	1								
	SHL	F	A+B	LD	1								1	0	1						0	0	1								
				SHL									1	1	1									0	1						
				SHL	1								1	1	0									0	1						
				NOP									1	1	1									1	1	1	0	1	0	0	1
		F	A+B	LD	1								1	1	1									1	1	1	0	1	0	0	1

Der Algorithmus „erstellt“ die roten Nullen erst im Nachhinein durch das nach links Shiften in jedem Schritt...